# 开发工具环境布局

开发工具：1、Adnroid-Adt-bundle 2、Android Studio

Adnroid-Adt-bundle 谷歌已经不支持下载sdk

ADT Bundle包含了Eclipse、ADT插件和SDK Tools，是已经集成好的IDE，只需安装好Jdk即可开始开发，推荐初学者下载ADT Bundle，不用再折腾开发环境

配置SDK环境步骤：

第一步：新建名称ANDROID\_HOME 值C:\adt-bundle-windows-x86\_64-20140702\sdk

第二步：修改Path,添加%ANDROID\_HOME%\tools;%ANDROID\_HOME%\build-tools;%ANDROID\_HOME%\platform-tools

第三步：打开DOS窗口，检查是否配置成功：输入 adb显示adb版本及命令帮助

输入android弹出窗口，成功

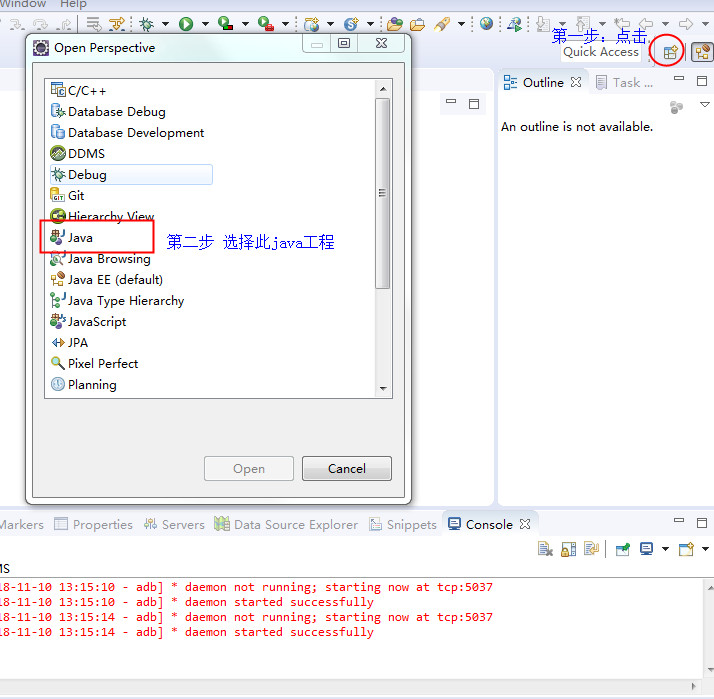
使用eclipse搭建环境

搭建eclipse环境(需要ADT的zip包)

第一步 下载ADT包：最新的好像是ADT-23.0.7，进行部署最新eclipse(2018年9月份)

配置离线ADT：Eclipse主窗口Help —> install New soft…. —> Add—> Archive —> 找到zip文件ADT-23.0.7 —>进行Add安装

第二步 安装好后：选择java工程才有安卓开发目录



第三步 eclipse选择SDK目录(系统配置SDK环境会自动选择)

window -🡪preferfences —>Android —>Browse.. —>找到自己的SDK目录即可

第四步 开发

右键 —>new —> AndroidApplicationProject

# UI布局控件

控件的父类View

通用属性：5种

1. 控件的宽高：（固定值 与两种状态）

android:layout\_width="30dp"

android:layout\_height="30dp"

2. 控件的可见度：（3种状态）

android:visibility="visible"

3. 控件的背景：（颜色 图片）

android:background="#F00" //RGB(红绿蓝): #000~#FFF; ARGB(透明度+红绿蓝): #0000~#FFFF

4. 盒子模型：

android:layout\_margin="10dp"

android:padding="5dp"

5. 点击事件：

android:onClick="haha"

## 布局(layout)

View与ViewGroup的关系：

1. View表示控件 View与View之间是无法嵌套

2. 为了让控件能够嵌套其他控件 才产生了ViewGroup

3.继承关系； ViewGroup一般比较少使用 我们一般使用其子类实现（也就是5大布局控件）

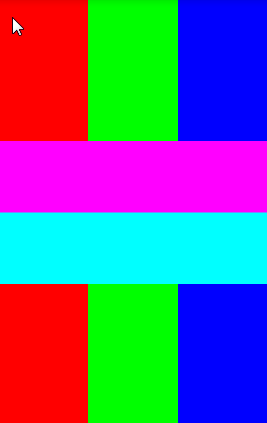
5大布局控件：

1、LinearLayout: 线性布局指的是该布局下包含的子布局列表为横线或者竖线排布。

指定方向：(横向布局/纵向布局) android:orientation="" 值为：vertica和horizontal

权重：android:layout\_weight="1" 指的是布局中：1比1形式，有三个布局这个属性把父容量分3比例

效果图



2、RelativeLayout: 相对布局 (相对父控件布局 相对兄弟控件布局)

针对父控件

android:layout\_centerHorizontal 横向居中

android:layout\_centerVertical 纵向居中

android:layout\_centerInParent 横向纵向居中

跟父控件顶部/底部/最左边/最右边对齐

android:layout\_alignParentTop

android:layout\_alignParentBottom

android:layout\_alignParentLeft

android:layout\_alignParentRight

针对已存在的兄弟控件(在某个控件的上面/下面/左边/右边)

android:layout\_above

android:layout\_below

android:layout\_toLeftOf

android:layout\_toRightOf

相对兄弟控件的边对齐

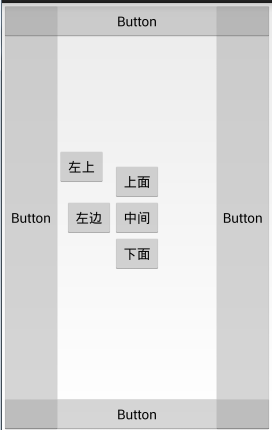
android:layout\_alignTop

android:layout\_alignBottom

android:layout\_alignLeft

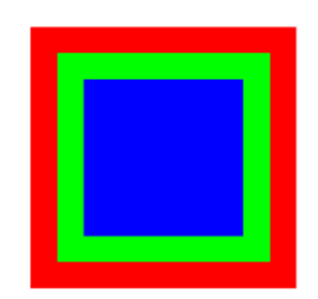
android:layout\_alignRight

效果图



3、FrameLayout: 帧布局 代码越是后面控件越在顶层 (应用场景: 霓虹灯) 层层叠加

效果图



4 AbsoluteLayout :绝对布局 控件在父控件中的绝对定位 (应用场景: 机顶盒开发)

android:layout\_x="30dp"

android:layout\_y="30dp

5、TableLayout: 表格布局 (应用场景: 银行表格)

也不怎么用，效果丑：一般都线性布局来代替

## 按钮和文本框

它们父类TextView:

android:text="文本"

android:textSize="20sp"

android:textColor="#FF0"

android:textStyle="bold"

android:lines="3"

android:singleLine="true"

android:typeface="monospace" //设置字型。字形有：normal, sans, serif,monospace

android:clickable=””

Button:

属性与TextView基本相似。

不同点：

1. 按钮是自带了背景的控件

2. 按钮是可以点击了

--------------------------------------------按钮有两种状态：按下和原来状态：1改变背景原色操作-----------------------------------------------

第一步 在res文件夹中创建一个drawable文件夹：xml类型Drawable选择：selector：

<selector xmlns:android="http://schemas.android.com/apk/res/android" >

<!-- button效果 有两种状态：1 原来状态，2.按下的状态 -->

<!-- 第0步把两张图片放入drawable-hdpi下 -->

<!-- 第一步 设置按下的状态 pressed加压 引入自己定义的按下图片-->

<item android:state\_pressed="true">

<shape >

<stroke android:color="#888" android:width="1dp"/>

<solid android:color="#FF0"/>

<corners android:radius="10dp"/>

</shape>

</item>

<!--注意默认的状态一定要写到最后面 第二步 设置原来的状态 引入自己定义的原来状态-->

<item>

<shape >

<!--stroke边框 width:粗细 -->

<stroke android:color="#888" android:width="1dp"/>

<!-- 输入框内填充的颜色 -->

<solid android:color="#FFF"/>

<!-- 设置圆角 -->

<corners android:radius="10dp"/>

</shape>

</item>

</selector>

---------------------------------------------------------------------2引入两张图片操作--------------------------------------------------------------------------

<selector xmlns:android="http://schemas.android.com/apk/res/android" >

<!-- button效果 有两种状态：1 原来状态，2.按下的状态 -->

<!-- 第0步把两张图片放入drawable-hdpi下 -->

<!-- 第一步 设置按下的状态 pressed加压 引入自己定义的按下图片-->

<item android:state\_pressed="true" android:drawable="@drawable/ic\_launcher"/>

<!--注意默认的状态一定要写到最后面 第二步 设置原来的状态 引入自己定义的原来状态-->

<item android:drawable="@drawable/ic\_launcher"/>

</selector>

---------------------------------------------------------------------文字颜色改变的操作 -------------------------------------------------------------------------

<selector xmlns:android="http://schemas.android.com/apk/res/android" >

<!-- button效果 有两种状态：1 原来状态，2.按下的状态 -->

<!-- 第一步 设置按下的状态 pressed加压 -->

<item android:state\_pressed="true" android:color="#00F"/>

<!--注意默认的状态一定要写到最后面 第二步 设置原来的状态 -->

<item android:color="#F00"/>

</selector>

---------------------------------------------------------------------上面三种方式引入操作 --------------------------------------------------------------------------

<Button <!-- 以背景原色引入图片的方式引入 -->

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_margin="2dp"

android:textColor="@drawable/btn\_text\_selector" //引入字体颜色的状态

android:text="按钮"

android:background="@drawable/btn\_selector" /> //引入上面前两个方式

EditText：

android:hint="请输入QQ"

android:editable="true"

android:maxLength="50" 设置最大的字数

android:inputType="textPassword"

进行优化操作：设置为圆弧的输入框

第一步 在res文件夹中创建一个drawable文件夹：xml类型Drawable选择：shape：

<shape xmlns:android="http://schemas.android.com/apk/res/android" >

<!--stroke边框 width:粗细 -->

<stroke android:color="#888" android:width="1dp"/>

<!-- 输入框内填充的颜色 -->

<solid android:color="#FFF"/>

<!-- 设置圆角 -->

<corners android:radius="10dp"/>

</shape>

当作上面的就是一个背景图片来引入：

<EditText

android:layout\_width="match\_parent"

android:layout\_height="32dp"

android:layout\_margin="2dp"

android:paddingLeft="5dp"

android:background="@drawable/et\_shap"/> //以背景引入图片操作

单选框

RadioButton：

<RadioGroup /> 单选组

<RadioButton /> 单选框

设置监听器的时候,通过 RadioGroup radioGroup.setOnCheckedChangeListener();

android:checked="true"

多选框

CheckBox：

设置监听器的时候 每个CheckBox都应该设置

android:checked="false"

优化：

<selector xmlns:android="http://schemas.android.com/apk/res/android" >

<!-- 一种是选中的 一种是未选中 -->

<item android:state\_checked="true"

android:drawable="@drawable/more\_radio\_selected"/>

<item

android:drawable="@drawable/more\_radio\_normal"/>

</selector>

引入：

<!-- CheckBox里面的选择框样式是由属性android:button决定的 -->

<CheckBox

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:button="@drawable/cbx\_slector"//引入操作

android:text="苹果" />

## WebView

小案例：让网页在此控件显示

需要注意上网权限问题和旋转重新加载页面两个问题

旋转重新加载页：在AndroidManifest.xml文件中标签<activity>中添加属性

android:configChanges="orientation|keyboardHidden|screenSize"。

上网权限问题：看第三步

//第一步 获取控件：view所有控件的父类强转

WebView wv = (WebView) findViewById(R.id.webView1);

//第二步 获得控件后进行加载页面

wv.loadUrl("http://106.12.106.21:8080/");

//第三步 此时是没有联网的权限需在AndroidManifest.xml加权限

// 这句话不在这里写：<uses-permission android:name="android.permission.INTERNET"/>

//第四步 启动js代码 ：默认是不支持JavaScript的

WebSettings settings = wv.getSettings();

settings.setJavaScriptEnabled(true);

//第五步 设置重定向的网页也在此控件显示 ： 一旦重定向跳转网页会导致跳转到系统的浏览器来显示，

wv.setWebViewClient(new WebViewClient(){

//返回true 意思是：不要让系统来操作帮忙跳转网页

//view:自己写的WebView控件，url：每次一点击都会获得新的网页地址传入

public boolean shouldOverrideUrlLoading(WebView view, String url) {

view.loadUrl(url);

return true;

}

});

}

AndroidManifest.xml文件的内容

<uses-sdk

android:minSdkVersion="23"

android:targetSdkVersion="23" />

<!-- 上网权限问题 -->

<uses-permission android:name="android.permission.INTERNET"/>

<application

android:allowBackup="true"

android:icon="@drawable/ic\_launcher"

android:label="@string/app\_name"

android:theme="@style/AppTheme" >

<activity

android:name=".Bill"

android:label="@string/app\_name"

<!-- 设置为旋转页面不重新加载 -->

android:configChanges="orientation|keyboardHidden|screenSize"

>

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>

## 进度条

优化：

第一步 在res文件夹中创建一个drawable文件夹：xml类型Drawable选择：layer-list：

<layer-list xmlns:android="http://schemas.android.com/apk/res/android" >

<!-- 进度条:背景图 进度图 -->

<item android:drawable="@drawable/progress\_bar\_bg"

android:id="@android:id/background"/>

<item android:drawable="@drawable/progress\_bar\_selected\_bg" //自己的图片

android:id="@android:id/progress"/>

</layer-list>

引入

<!-- 引用上面的进度条样式图 -->

<ProgressBar

style="?android:attr/progressBarStyleHorizontal"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:progressDrawable="@drawable/pb\_bg"

android:max="100"

android:progress="100" />

## ListView

### 使用步骤

第一步 MainActivity中代码

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

//第一步 ListView获取列表容器

ListView lv = (ListView) findViewById(R.id.lv);

ArrayList<String> list= new ArrayList<String>();//测试：准备好数据

for (int i = 0; i < 30; i++) {

list.add("测试："+i);

}

//第二步 因数据不是直接交互的是：适配器作为中间来处理数据

MyBaseAdapter ba= new MyBaseAdapter(list); //数据放入适配器中

lv.setAdapter(ba); //把数据和容器关联

}

第二步 BaseAdapter适配器类

/\*\*

\* 注意：每次进入此类系统都会默认调用 getCount()方法 和 getView方法

\* @author Administrator

\*/

public class MyBaseAdapter extends BaseAdapter {

private ArrayList<String> list;

public MyBaseAdapter(ArrayList<String> list) {

this.list=list;

}

//第一步 此方法每次进入此类 系统都会调用：用来获取数据的长度

public int getCount() {

return list.size(); //把数据返回

}

//第二步 此方法每次进入此类系统都会调用：用来获取View

/\*\*

\* position :返回当前标签的索引

\* convertView : 缓存View：老师测试：刚进来加载显示:为null，虚拟机显示‘测试0’遮住后就会有缓存测试0的数据

\* parent：有值的view：主要用来获取context

\*/

public View getView2(int position, View convertView, ViewGroup parent) {

//第三步 创建标签TextView 有两种方式：

if(convertView ==null) {

// 第一种 方式使用代码来创建一个标签TextView

TextView tv = new TextView(parent.getContext());

//查看源码LayoutParams是ViewGroup的内部类,很多类都有这内部类继承了ViewGroup的内部类LayoutParams

tv.setLayoutParams(new AbsListView.LayoutParams(//设置宽和高

AbsListView.LayoutParams.MATCH\_PARENT, AbsListView.LayoutParams.WRAP\_CONTENT));

tv.setTextSize(30); //设置字体大小

}

tv.setText(list.get(position)); //TextView的名称

//第二种方式 先在layout文件夹中创建xml 根为TextView标签

//(1)布局充气的一个类 就是可以将一个布局转换成一个View

LayoutInflater layoutInflater = LayoutInflater.from(parent.getContext());

//(2)创建所有标签View的父类：参数1：就是layout你创建的Xml，根标签是TextView，参数2是否需要添加到某个容器里面去

View view = layoutInflater.inflate(R.layout.textview\_layout, null);

//(3)通过findViewById获取此xml：你在layout创建的根标签TextView中设置了id为tv2

TextView txv = (TextView) view.findViewById(R.id.tv2);

txv.setTextSize(30);

txv.setText(list.get(position));

return txv;

}

public Object getItem(int position) {

return null;

}

public long getItemId(int position) {

return 0;

}

调用android系统中自带的TextView标签

//(1)布局充气的一个类 就是可以将一个布局转换成一个View

LayoutInflater layoutInflater = LayoutInflater.from(parent.getContext());

//(2)创建所有标签View的父类：参数1：系统自带的layout布局 2是否需要添加到某个容器里面去

convertView = layoutInflater.inflate(android.R.layout.simple\_list\_item\_2,null);

//(3)通过findViewById获取此xml：你在layout创建的根标签TextView中设置了id为tv2

TextView txv = (TextView) convertView.findViewById(android.R.id.text1);//获取第一个TextView

TextView txv2 = (TextView) convertView.findViewById(android.R.id.text2);//获取第二个TextView

## 通用属性

这里介绍了：

1、图片 2、滚动条3、日期和时间选择器

1、图片

设置图片源

android:src="@drawable/ic\_launcher"

代码设置图片源

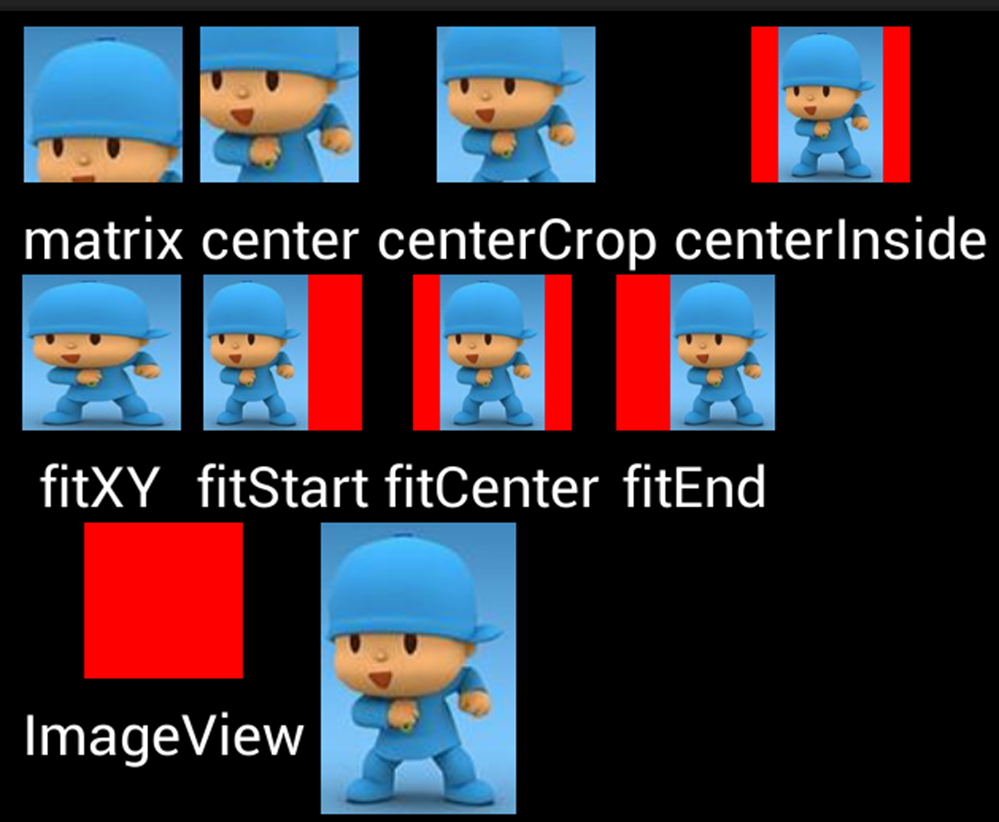
imageView.setImageResource(resId)

imageView.setImageBitmap(bm);

imageView.setImageDrawable(drawable);

设置缩放模式

android:scaleType="" 下面图片是属性值和效果图



4、滚动条使用

1. 手机界面无法容纳更多的控件的时候，就需要滚动界面。

2. ScrollView :控制上下滑动的效果

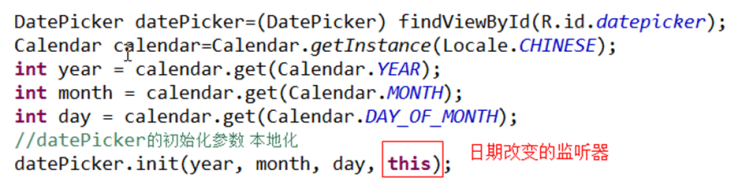
3. HorizontalScrollView: 控制左右滑动的效果

4. ScrollView与HorizontalScrollView只能容纳一个子控件。

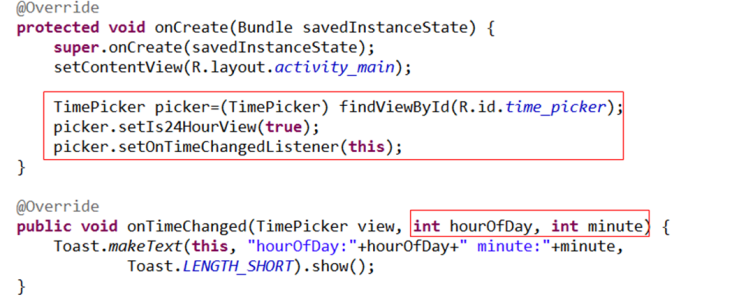


5、日期和时间选择器

**1. 日期选择器:**



**2. 时间选择器: 24小时制**



# android存储

保存数据到文件中有两个目录

files目录：路径:/data/data/应用的包名/files

此目录一般保存的是重要的信息 用户信息 应用的配置

清除此目录保存数据的操作：设置🡪 应用程序🡪应用详情🡪清除数据(删除cache目录 删除files目录)

代码获取files目录方法：//getFilesDir()获取file文件夹的路径 File file = new File(getFilesDir(), "/filelog.txt");

注意 测试了两台手机一台 4.1.2 一台 6.0 。调用 Context.getFilesDir.getAbsolutePath 方法。

4.1.2 返回 /data/data/package/files

6.0 返回 /data/user/0/package/files)

6.0系统以后是 /data/user/x/package/files x 代表用户号 6.0 支持多用户

保存的数据都是只有自己才能读写的：如果希望别的应用可以访问需：

File file = new File("/data/data/com.m520it.qqlogin/qqinfo.txt");

// 把文件保存到cache目录下

File file = new File(getFilesDir(), "/qqinfo.txt");

// openFileOutput：两个参数name 就是文件名，mode 文件权限

// MODE\_APPEND 比如一个文件已经有内容了 再次写文件就是在文件后面继续追加

// MODE\_PRIVATE 对其他应用私有化 但是本应用是可读可写的

// MODE\_WORLD\_READABLE 任何应用都能读取

// MODE\_WORLD\_WRITEABLE 任何应用都能写入

FileOutputStream fos=openFileOutput("qqinfo.txt",

MODE\_WORLD\_READABLE+MODE\_WORLD\_WRITEABLE);

BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(fos));

writer.close();

cache目录：路径:/data/data/应用的包名/cache

此目录可以用来缓存不重要的数据 保存图片 等

清除此目录保存数据的操作：设置🡪 应用程序🡪 应用详情🡪 清除数据(删除cache目录下的文件)

代码获取cache目录方法：//getFilesDir()获取file文件夹的路径 File file = new File(getCacheDir(), "/filelog.txt");

## sqllite数据库

概念

创建数据库：

具有读写权限

生成目录：位于data/data/应用程序包名/databases文件目录

SQLite中创建表

SQLite中所有数据类型默认存储都是字符串类型的，没有长度限制的。超过表定义时的数据长度，也不会报错。

在Android中SQLite会自动创建一张名为android\_metadata的表。用于存储当前的语言环境的。在SQLite中id推荐使用\_id。

SQLite的数据库版本只能增大，不能减小。

游标：在访问数据库中表结构时，想访问表中的某一行的时候，数据库内部有一个快速的定位方式，这个定位方式是通过索引来实现的。游标相当于数组的指针，通过游标的上下移动来查找数据

创建数据表步骤： 本身已内置了sqlite，不用导包之类操作

第一步 创建数据库和表：继承抽象类 SQLiteOpenHelper实现两个方法

//第一步 继承抽象类 SQLiteOpenHelper 实现它的构造器(2个构造器是4个和5个参数)、两个方法

public class DbOpenHelper extends SQLiteOpenHelper {

//数据库文件名

public static final String DB\_NAME="wwl.db";

//数据库版本

public static final int DB\_VERSION=2;

//抽取出数据库的表名和字段(需要在dao类中用到)

public static final String DB\_TABLENAME="AAA"; //表名

public static final String COLUMN\_ID="\_id"; //字段

public static final String COLUMN\_NAME="name"; //字段

public static final String COLUMN\_PHONE="phone"; //字段

public static final String COLUMN\_MONEY="money"; //字段

//第二步 (可以省略)父类构造：参数较多使用本类构造器只定义一个参数的：Context，其它参数本类定义

/\*\*

   \* @param context 上下文

    \* @para mname  数据库文件的名称

    \* @param factory 用来创建游标对象，null就用默认的游标工厂

    \* @param version 数据库的版本 号，从1开始。用来更新数据库。

    \* 数据库后缀名不是固定的，写为.db只是为了方便查看和识别

  \*/

public DbOpenHelper(Context context) {

super(context, DB\_NAME, null, DB\_VERSION);

}

//第三步 onCreate方法的作用：什么时候会调用

//数据库第一次创建的时候自动执行,如果数据库已经存在，则不会再次调用该方法

//主要用来创建数据库表操作

public void onCreate(SQLiteDatabase db) {

//执行创建表的SQL语句

db.execSQL("create table "+DB\_TABLENAME+"("+COLUMN\_ID+" integer primary key autoincrement,"

+COLUMN\_NAME+" varchar(20),"+COLUMN\_PHONE+" varchar(20))");

}

//第三步 onUpgrade方法的作用：什么时候会调用

//数据库版本更新时自动执行,即数据库版本号有变化的时候执行 数据库的版本只能变大，不能变小

//int oldVersion, int newVersion用来完成跨版本升级数据库时保持数据

//如果是垄断性行业，数据是一次有效的情况下，可以强制以前数据失效的方式升级，让用户重新输入数据

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

//新版本需要向表中添加一个字段money

db.execSQL("alter table "+DB\_TABLENAME+" add "+COLUMN\_MONEY+" varchar(10)");

}

第二步 调用操作

//执行该行代码，数据库不会被创建，只是创建了一个数据库对象

DbOpenHelper db =new DbOpenHelper(this);

//创建或者打开数据库，该数据库具有读写权限，位于data/data/应用程序包名/databases/文件夹下面

SQLiteDatabase sd = db.getWritableDatabase();

//准备sql语句

        String sql = "insert into info(name,phone)values(?,?)";

        //在SQLite中，也可以使用占位符来传递参数

        db.execSQL(sql, new Object[]{"lisi","453231"});

        //释放资源

        db.close();

}

封装到dao包中操作

第三步 增删改查的操作(CRUD)

public class ContactDao {

//创建上面自己写的类:通过构造器进行调用

private DbOpenHelper mHelper;

public ContactDao(Context c) {

mHelper=new DbOpenHelper(c);

}

//插入操作

public boolean insertContact(String username,String phone){

//第一次会创建或者打开数据库，该数据库具有读写权限，位于data/data/应用程序包名/databases/文件夹下面

SQLiteDatabase db = mHelper.getWritableDatabase();

//ContentValues 键值对 源代码里面只是简单的维护了一个HashMap

ContentValues values=new ContentValues();

//key 插入的数据库表的字段名 value就是要插入的值

values.put(DbOpenHelper.\_USERNAME, username);

values.put(DbOpenHelper.\_PHONE, phone);

//参数：第一个表名，第二个是表示nullColumnHack表示是否为空，第三个values表示键值对

//nullColumnHack 比如 放了username 那么插入的时候 username就不能为null

//如果nullColumnHack设置为null 那么此次插入的字段可以允许某些字段不插入

long rowId = db.insert(DbOpenHelper.TABLE\_NAME, null , values);

//@return 新插入的行的id 如果是-1则说明插入失败

return rowId != -1;

}

//修改操作

public boolean updateContact(String username,String newPhone){

SQLiteDatabase db = mHelper.getWritableDatabase();

ContentValues values=new ContentValues(); //设置键值对象

values.put(DbOpenHelper.\_PHONE, newPhone); //放入字段名和值

//参数1：表名，参数2：表示更新的键值对，参数3：where后面语句，参数4：where语句绑定的值

int updateRows = db.update(DbOpenHelper.TABLE\_NAME,

values, DbOpenHelper.\_USERNAME+"=?", new String[]{username});

//@return int 更新了几行就返回多少

return updateRows>0;

}

//删除操作

public boolean deleteContact(String username){

SQLiteDatabase db = mHelper.getWritableDatabase();

//whereClause where语句

//参数1：表名，参数2：where后面语句，参数3：where语句绑定的值

int deleteRows = db.delete(DbOpenHelper.TABLE\_NAME, "username=?", new String[]{username});

return deleteRows>0;

}

//查询操作

public String queryContact(String searchPhone){

SQLiteDatabase db = dbOpenHelper.getReadableDatabase();

//table 表示表名

//columns 返回多少列 比如 new String[]{'列名','列名2'} 行数

//selection where语句

//selectionArgs表示绑定的值

Cursor cursor = db.query(DbOpenHelper.TABLE\_NAME,new String[]{DbOpenHelper.USERNAME,DbOpenHelper.PHONE} ,

""+dbOpenHelper.PHONE+"=?", new String[]{phone}, null, null, null);

String result="";

while(cursor.moveToNext()){

//根据列名返回索引

int usernameIndex=cursor.getColumnIndex(DbOpenHelper.USERNAME);

String username = cursor.getString(usernameIndex);

int phoneIndex = cursor.getColumnIndex(DbOpenHelper.PHONE);

String phones = cursor.getString(phoneIndex);

result+=("名字 : "+username+" 电话 : "+phones+" \n");

}

return result;

}

注意：在这里有   SQLiteOpenHelper对象.getWritableDatabase();或getReadableDatabase();两种的区别在

1.getWritableDatabase();用于增删改

        --用于写数据比如你要插入一条数据插入进去是写数据    写数据就要使用getWritableDatabase();方法返回一个SQLiteDatabase对象

2.getReadableDatabase();用于查询

            --用于读数据比如你要查看一条数据就是读在数据表中读取数据进来    写数据就要使用getReadableDatabase();方法返回一个SQLiteDatabase对象

## 内部存储

路径：/data

代码：获取方法ENVironment.getDataDirectory()

File dataFile=ENVironment.getDataDirectory();

Log.v(“路径：”,dataFile.getAbsolutePath())

## 外部存储(SD卡)

路径：根据手机型号不同而不同

代码：获取方法ENVironment.getDataDirectory() //调用此方法都能获取不同厂商的路径

File dataFile=ENVironment.getDataDirectory();

Log.v(“路径：”,dataFile.getAbsolutePath())

SD卡读写操作

1. 将文件写入到SD卡上首先要判断SD卡是否装载

//1.SD卡是可以插拔的 判断下SD卡是否装载

String externalState = Environment.getExternalStorageState(); //获取一个状态

if (externalState.equals(Environment.MEDIA\_MOUNTED)) { // 判断是否MOUNTED(安装)

File parentFile = Environment.getExternalStorageDirectory();

File file=new File(parentFile, "葫芦娃.avi");

//字节流

FileOutputStream fos=new FileOutputStream(file);

byte[] buff=new byte[1024\*1024];

for (int i = 0; i < 10; i++) {

fos.write(buff);

}

fos.close();

2、因为读写SD卡是敏感的行为 需要用户同意所以需要使用权限

找到AndroidMenifest.xml文件，在后面添加使用权限：

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"/>

## SharedPreferences

使用SharedPreferences保存key-value对的步骤如下：

（1）使用Activity类的getSharedPreferences方法获得SharedPreferences对象，其中存储key-value的文件的名称由getSharedPreferences方法的第一个参数指定，第二个参数指定访问应用程序私有文件的权限。

（2）使用SharedPreferences接口的edit获得SharedPreferences.Editor对象。

（3）通过SharedPreferences.Editor接口的putXxx方法保存key-value对。其中Xxx表示不同的数据类型。例如：字符串类型的value需要用putString方法。

（4）通过SharedPreferences.Editor接口的commit方法保存key-value对。commit方法相当于数据库事务中的提交（commit）操作。

a、打开Preferences，名称为setting，如果存在则打开它，否则创建新的Preferences

SharedPreferences userSettings = getSharedPreferences("setting", 0); //参数：文件名，不存在的返回值

b、让setting处于编辑状态

SharedPreferences.Editor editor = userSettings.edit();

c、存放数据

editor.putString("name","ATAAW");

editor.putString("URL","ATAAW.COM");

d、完成提交

editor.commit();

读取数据信息，步骤如下：

　a、获取Preferences

SharedPreferences userSettings= getSharedPreferences("setting", 0);

b、取出数据

String name = userSettings.getString("name","默认值");

String url = userSettings.getString("URL","default");

删除和清空操作，如下：

a、清除指定数据

SharedPreferences.Editor editor = userSettings.edit();

editor.remove("KEY");

editor.commit();

b、清空数据

SharedPreferences.Editor editor = userSettings.edit();

editor.clear();

editor.commit();

注意：如果在 Fragment 中使用SharedPreferences 时，需要放在onAttach(Activity activity)里面进行SharedPreferences 的初始化，否则会报空指针 即 getActivity()会可能返回null ！